SDM COLLEGE OF ENGINEERING AND TECHNOLOGY

Dhavalagiri, Dharwad-580002, Karnataka State, India.

Email: cse.sdmcet@gmail.com

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

A Report
on

DBMS – Minor Assignment

COURSE CODE: 22UCSC501
COURSE TITLE: Database Management System
SEMESTER: 5   DIVISION: A
COURSE TEACHER: Dr. U P Kulkarni

[Academic Year- 2024-25]

Date of Submission: 24-10-2024

Submitted
By

Mr. Venkatesh Rao S
USN: 2SD23CS411

<u>Table of Contents</u>

Minor Work:

A1:Write a C program to study all file operations related SYSTEM CALLS supported
        byUNIX OS and C libraries for file operations.

```c
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>
#include <string.h>
#include <errno.h>

#define FILENAME "dbms.txt"
#define BUFFER_SIZE 100

int main() {
   int fd;  // File descriptor
   char text[] = "Hello, this is a test file.\n";  // Data to write to the file
   char buffer[BUFFER_SIZE];  // Buffer to hold read data

   // 1. Create and open a file for writing
   fd = open(FILENAME, O_CREAT | O_WRONLY | O_TRUNC);
   if (fd == -1) {
      perror("Error opening file for writing");
      return EXIT_FAILURE;
   }
   printf("File '%s' created successfully.\n", FILENAME);

   // 2. Write to the file
   if (write(fd, text, strlen(text)) == -1) {
      perror("Error writing to file");
      close(fd);
      return EXIT_FAILURE;
   }
   printf("Data written to file successfully.\n");

   // 3. Close the file
   if (close(fd) == -1) {
      perror("Error closing file after writing");
      return EXIT_FAILURE;
   }
   printf("File closed successfully after writing.\n");
```

```c
// 4. Open the file for reading
fd = open(FILENAME, O_RDONLY);
if (fd == -1) {
   perror("Error opening file for reading");
   return EXIT_FAILURE;
}
printf("File '%s' opened for reading.\n", FILENAME);

// 5. Read from the file
ssize_t bytesRead = read(fd, buffer, sizeof(buffer) - 1);
if (bytesRead == -1) {
   perror("Error reading from file");
   close(fd);
   return EXIT_FAILURE;
}
buffer[bytesRead] = '\0'; // Null-terminate the buffer
printf("Data read from file: %s", buffer);

// 6. Close the file after reading
if (close(fd) == -1) {
   perror("Error closing file after reading");
   return EXIT_FAILURE;
}
printf("File closed successfully after reading.\n");

// 7. Delete the file
if (remove(FILENAME) == 0) {
   printf("File '%s' deleted successfully.\n", FILENAME);
} else {
   perror("Error deleting file");
}

   return EXIT_SUCCESS;
}
```

Output:

```
PS C:\Users\ABDULBASITH-HOME> cd "c:\Users\ABDULBASITH-HOME\Documents\" ; if ($?) {
      gcc dbmsA1.c -o dbmsA1 } ; if ($?) { .\dbmsA1 }
File 'dbms.txt' created successfully.
Data written to file successfully.
File closed successfully after writing.
File 'dbms.txt' opened for reading.
Data read from file: Hello, this is a test file.
File closed successfully after reading.
File 'dbms.txt' deleted successfully.
PS C:\Users\ABDULBASITH-HOME\Documents>
```

A2:Write a C program to demonstrate indexing and associated operations.

```c
#include <stdio.h>
#include <sqlca.h>
EXEC SQL INCLUDE SQLCA;

int main() {
  // Connect to the database
  //EXEC SQL CONNECT :username IDENTIFIED BY :password;
  EXEC SQL CONNECT :22cs001 IDENTIFIED BY :a;

  if (sqlca.sqlcode != 0) {
    printf("Error connecting to the database: %d\n", sqlca.sqlcode);
    return 1;
  }

  printf("Connected to the database.\n");

  // Create a table
  EXEC SQL EXECUTE IMMEDIATE "CREATE TABLE employees (emp_id NUMBER PRIMARY
      KEY, emp_name VARCHAR2(50), emp_dept VARCHAR2(30), emp_salary NUMBER)";

  if (sqlca.sqlcode != 0) {
    printf("Error creating table: %d\n", sqlca.sqlcode);
    return 1;
  }
  printf("Table created successfully.\n");

  // Insert data
  EXEC SQL EXECUTE IMMEDIATE "INSERT INTO employees (emp_id, emp_name, emp_dept,
      emp_salary) VALUES (101, 'AbdulBasith', 'HR', 50000)";

  if (sqlca.sqlcode != 0) {
    printf("Error inserting data: %d\n", sqlca.sqlcode);
    return 1;
  }
  printf("Data inserted successfully.\n");

  // Create an index
  EXEC SQL EXECUTE IMMEDIATE "CREATE INDEX emp_dept_idx ON employees(emp_dept)";

  if (sqlca.sqlcode != 0) {
    printf("Error creating index: %d\n", sqlca.sqlcode);
    return 1;
  }
  printf("Index created successfully.\n");
```

```
// Query the data (with index)
EXEC SQL EXECUTE IMMEDIATE "SELECT * FROM employees WHERE emp_dept = 'HR'";

if (sqlca.sqlcode != 0) {
    printf("Error querying data: %d\n", sqlca.sqlcode);
    return 1;
}

// Drop the index
EXEC SQL EXECUTE IMMEDIATE "DROP INDEX emp_dept_idx";

if (sqlca.sqlcode != 0) {
    printf("Error dropping index: %d\n", sqlca.sqlcode);
    return 1;
}
printf("Index dropped successfully.\n");

// Commit and disconnect
EXEC SQL COMMIT WORK;
EXEC SQL DISCONNECT;

printf("Disconnected from the database.\n");
return 0;
}
```

Indexing with only SQL commands:

/*Create a table*/

Create table employee
(
      empno integer not null
      constraint EMPLOYEE_PK_VIOLATION
      primary key,
      empname char(20) not null,
      sex char(1) not null
      Constraint EMPLOYEE_SEX_VIOLATION
      check (sex in ('m','f')),
      phone integer null,
      dob date default '01-jan-1901' not null
);

/*Insert data into table*/

Insert into employee values(&eno,&ename,&sex,&phone,&dob);

/*Create a Index on sex*/

Create INDEX empsex_index on employee(sex);

/*Query the data*/

Select * From employee where sex= 'm';

/*View the query execution plan*/

EXPLAIN PLAN FOR
SELECT * FROM employeesWHERE sex = 'm';
SELECT * FROM table(DBMS_XPLAN.DISPLAY);

Output:

SQL> Create INDEX empsex_index on employee(sex);

      Index created.

SQL> select * from employee where sex='m';

```
   EMPNO EMPNAME          S    PHONE       DOB
---------- ------------------- - ---------- ------------- --------------
     1 abdulbasith        m    9035789789  05-MAY-03
     2 bhanu              m    4165        19-MAY-03
     3 abhishek           m    897         12-DEC-04
     4 joy                m    732         18-AUG-05
     5 pratham            m    46          17-JUN-12
     6 lalit              m    7896        14-SEP-05
     7 akhil              m    135         14-AUG-02
     8 sagar              m    5285        03-DEC-07
     9 chinmay            m    45625       23-NOV-12
```

SQL> Explain plan for

  2 select * from employee where sex='m';

Explained.

SQL> SELECT * FROM table(DBMS_XPLAN.DISPLAY);

PLAN_TABLE_OUTPUT

Plan hash value: 4021620938

| Id | Operation | Name | Rows | Bytes | Cost (%CPU)| Time |

PLAN_TABLE_OUTPUT

| 0 | SELECT STATEMENT | | 6 | 228 | 2 (0) | 00:00:01 |

| 1 | TABLE ACCESS BY INDEX ROWID| EMPLOYEE | 6 | 228 | 2 (0) | 00:00:01 |

|* 2 | INDEX RANGE SCAN | EMPSEX_INDEX | 6 | | 1 (0) | 00:00:01 |

PLAN_TABLE_OUTPUT

Predicate Information (identified by operation id):

----

  2 - access("SEX"='m')

14 rows selected.

<u>A3</u>:Write a Java program to access the given excel file with known file format.

```java
package dbms123;
import java.io.File;
import java.io.FileInputStream;
import java.util.Iterator;
import org.apache.poi.xssf.usermodel.XSSFSheet;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;
import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.Row;

public class ReadExcel {
   public static void main(String[] args) {
      try {
         FileInputStream file = new FileInputStream(new File("input.xlsx"));
         XSSFWorkbook workbook = new XSSFWorkbook(file);
         XSSFSheet sheet = workbook.getSheetAt(0);
         Iterator<Row> rowIterator = sheet.iterator();

         while (rowIterator.hasNext()) {
            Row row = rowIterator.next();
            Iterator<Cell> cellIterator = row.cellIterator();

            while (cellIterator.hasNext()) {
               Cell cell = cellIterator.next();

               switch (cell.getCellType()) {
                  case NUMERIC:
                     System.out.print(cell.getNumericCellValue() + "\t");
                     break;
                  case STRING:
                     System.out.print(cell.getStringCellValue() + "\t");
                     break;
                  default:
                     System.out.print("Unknown type\t");
                     break;
               }
            }
            System.out.println("");
         }
         file.close();
         workbook.close();
      } catch (Exception e) {
         e.printStackTrace();
      }
   }
}
```